韓國 컴퓨터情報學會 論文誌
第13卷 第1號, 2008. 1.

2008-13-1-1-1

의사결정 트리 기법을 이용한 그리드 자원선택 시스템

노 창 현*, 조 규 철*, 마 용 범*, 이 종 식**

Grid Resource Selection System Using Decision Tree Method

Chang Hyeon Noh*, Kyu Cheol Cho*, Yong Beom Ma*, Jong Sik Lee**

요 약

이 기종의 네트워크와 시스템 자원으로 구성된 그리드 컴퓨팅 환경에서 대용량 데이터를 빠르고 정확하게 처리하기 위해서는 효과적인 그리드 자원선택이 필수적이다. 이를 위해 본 논문은 의사결정 트리 기법을 이용한 그리드 자원선택 시스템을 제안한다. 이 시스템은 자원 정보를 기록한 데이터 셋을 바탕으로 사용자들이 선택하는 자원들을 처리 할 데이터의 특성과 사용자의 요구사항으로 분석해서 자원선택을 위한 의사결정 트리를 구축한다. 그리드 사용자의 자원 요청 시 의사결정 트리를 탐색하여 사용자 요구 및 작업 특성에 적합한 자원들을 선택하여 작업을 할당함으로써 사용자 만족도를 향상시킴은 물론 전체 그리드 시스템의 성능을 개선한다. 실험결과는 본 논문에서 제안한 의사결정 트리 기반의 그리드 자원선택 시스템이 기존 그리드 자원선택 시스템인 Condor-G 및 Nimrod-G와 비교하여 더 높은 작업 처리율 및 자원 이용률과 더 적은 작업 손실 및 처리시간을 제공함으로써 그리드 자원선택 및 데이터 분산 처리에 효과적이라는 사실을 증명한다.

Abstract

In order to high-performance data processing, effective resource selection is needed since grid resources are composed of heterogeneous networks and OS systems in the grid environment. In this paper, we classify grid resources with data properties and user requirements for resource selection using a decision tree method. Our resource selection method can provide suitable resource selection methodology using classification with a decision tree to grid users. This paper evaluates our grid system performance with throughput, utilization, job loss, and average of turn-around time and shows experiment results of our resource selection model in comparison with those of existing resource selection models such as Condor-G and Nimrod-G. These experiment results showed that our resource selection model provides a vision of efficient grid resource selection methodology.

▸ Keyword : 의사결정 트리(Decision Tree), 그리드 자원선택(Grid Resource Selection), QoS

• 제1저자 : 노창현
• 접수일 : 2007. 11. 9, 심사일 : 2007. 12. 3, 심사완료일 : 2008. 1. 25.
* 인하대학교 정보공학과, ** 인하대학교 정보공학과 부교수

Ⅰ. 서 론

이기종의 시스템으로 구성된 그리드 컴퓨팅 환경에서 고성능의 데이터 처리 능력을 얻기 위해서는 어떠한 자원에서 데이터 처리가 수행 될 것인지가 중요한 문제이다. 또한, 그리드 사용자들은 데이터의 종류와 특성에 따라서 요구하는 자원이 다를 수 있으며 지불 가능한 금액[1]과 어플리케이션이 완전히 실행이 될 때까지의 데드라인 시간 등 요구사항이 각기 다를 수 있다.

효과적으로 그리드 자원을 이용하기 위해서는 사용자가 처리하려는 데이터의 특성에 따라 자원을 검색하고 선택 할 수 있어야 한다. 예를 들어 데이터의 크기는 상대적으로 작지만, 빠른 계산 능력을 요구하며 예산이 충분한 사용자의 경우 비싸더라도 고성능의 CPU들이 탑재 되어있고, 시간 안에 주어진 문제를 해결 할 수 있는 고가의 그리드 자원을 선호 할 수 있다. 만약, 이 사용자에게 적합하지 않은 자원을 검색 및 선택하게 되면, 오랜 검색시간과 함께 시간이나 비용 등 원하지 않는 처리결과로 자원의 재검색 및 선택에 따른 불만족스러운 QoS(Quality of Service)와 전체 그리드 컴퓨팅의 성능 저하를 초래 할 수 있다. 따라서 각각의 사용자마다 처리하고자 하는 데이터의 특성을 분석 후 분류하여 적절한 자원을 선택하면 그리드 컴퓨팅의 전반적인 시스템 성능 향상과 QoS를 향상 시킬 수 있다.

본 논문에서는 그리드 사용자에게 적절한 자원을 검색 및 선택 할 수 있도록 데이터 마이닝 기법 중 하나인 의사결정 트리를 이용하여 사용자의 데이터 특성과 요구사항을 기반으로 선택 자원을 분석 및 분류하고 자원 목록을 생성한다. 의사결정 트리[2]는 적은 양의 데이터로 비교적 정확한 분석 및 분류가 가능하며 대규모 데이터 집합의 시스템으로의 확장도 용이하고 모형 구축 시간이 짧은 장점이 있다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구의 기초가 되는 그리드 자원선택 시스템과 의사결정 트리 기법에 대해 소개하고, 3장에서는 본 논문에서 제안하는 의사결정 트리 기법을 이용한 그리드 자원선택 시스템을 설명한다. 그리고 4장에서 시뮬레이션을 통해 성능을 평가하고 5장에서 결론을 맺는다.

Ⅱ. 관련 연구

2.1 그리드 자원선택 시스템

그리드 컴퓨팅 환경에서 자원들은 각기 운영 체제와 CPU 개수, 처리 속도, RAM의 크기, 저장 매체의 크기가 매우 다양하다. 그리드 사용자들로부터 처리를 요청받은 데이터 또한 빠른 처리 속도를 원하는 계산위주의 데이터와 수십 수백 기가바이트(GByte)에 이르는 대용량의 데이터와 같이 그 특성이 다양하다. 이처럼 다양한 특성이 있는 사용자의 데이터 특성과 요구사항[3, 4]을 만족시키기 위해서는 해당 데이터의 처리에 적합한 자원을 검색 및 선택하는 시스템이 필요하다.

현재 그리드 상의 자원을 선택 할 수 있는 기능을 제공하는 대표적인 그리드 미들웨어로는 글로버스 툴킷(Globus Toolkit)과 Condor-G, Nimrod-G 등이 있다. 글로버스 툴킷[5]은 MDS(Metacomputing Directory Service)를 이용하여 사용자가 XML 형태로 질의해서 자원에 대한 정보를 얻을 수 있으나 자원선택에 관해서는 별다른 방안을 제시하고 있지 않다. Condor-G[6]는 자원에 대한 플랫폼과 메모리 사용률에 따른 가용도를 측정하여 사용자가 요구하는 자원수와 매칭 해주는 역할을 한다. Nimrod-G[7]는 사용자의 예산과 작업의 종료기한 조건으로 자원을 검색 및 할당한다. 하지만 이러한 기존의 그리드 미들웨어는 자원선택 시 사용자의 데이터 특성과 요구사항을 충분히 반영하지 못하고 있다. 따라서 본 논문에서는 사용자의 데이터 특성과 요구사항을 기반으로 선택 가능한 자원들을 샘플 데이터로 저장한 후, 의사결정 트리 기법을 이용하여 이를 분석하고 분류한다. 분류된 자료로부터 그리드 사용자의 자원사용 요청 시 자동적인 자원 검색 및 선택을 가능케 하여 사용자의 QoS를 향상 시킬 수 있다.

2.2 의사결정 트리(Decision tree)

의사결정 트리[8]는 데이터마이닝에서 분류 작업에 사용되는 기법으로, 이전에 저장된 데이터의 레코드들을 분석하여 이들 사이에 존재하는 패턴, 즉 부류별 특성을 속성의 조합으로 나타내는 분류모형 트리의 형태로 만드는 것이다. 만들어진 분류모형은 새로운 레코드를 분류하고 해당 부류의 값을 예측하는데 사용된다. 의사결정 트리를 만드는 알고리즘으로는 그림 1과 같은 C4.5 알고리즘[9]이 이용된다.

```
Function decision_tree(example_set, Properties(Attribute))
Begin
 if all entries in example_set are in the same class
  then return a leaf node labeled with that class
 else if Properties is empty
  then return leaf node labeled with disjunction of all classes
 else begin
  select a property, P, to text on and make it root of current tree;
  delete P from properties;
  for each value, V, of P,
  begin
   create a branch of the tree labeled with V
   let partition be elements of example_set with V for property P;
   call induce_tree(partition, properties), attach result to branch V
  end;
 end;
End;
```

그림 1. C4.5 알고리즘
Fig. 1. C4.5 Algorithm

C4.5 알고리즘[10]은 J Ross Quinlan에 의해 ID3를 수정 발전시킨 의사결정 알고리즘이다. 샘플을 분류하는데 정보 이득(Information Gain)이라는 엔트로피(Entropy)를 이용하여 분리한다. p_i를 샘플이 클래스 C_i에 속할 확률로 s_i / s 으로 계산 했을 때, 주어진 샘플 S에 있는 s_i개 샘플들을 분류하는데 요구되는 기대 정보량(Expected Information)은 수식 (1)과 같다.

$$I(s_1,s_2,...,s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \text{ [8]} \cdots\cdots\cdots (1)$$

속성 A가 v개의 다른 값을 갖는다면 A는 S를 부분집합으로 분할하는데 사용될 수 있다. 부분 집합 S_j는 A의 값 a_{ij}를 갖고 클래스 C_i의 샘플 개수를 s_{ij}라고 했을 때, A에 의해 분할 될 경우의 엔트로피 기대정보량은 수식 (2)와 같다.

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j}+...+s_{mj}}{s} I(s_{1j},...,s_{mj}) \text{ [8]} \cdots\cdots\cdots (2)$$

S_j의 샘플이 클래스 C_i에 속할 확률을 $p_{ij} = s_{ij}/|S_j|$로 정의하면, 수식 (3)을 이용하여 부분 집합 S_j의 기대정보량을 계산할 수 있다.

$$I(s_1,s_2,...,s_m) = -\sum_{i=1}^{m} p_{ij} \log_2(p_{ij}) \text{ [8]} \cdots\cdots\cdots (3)$$

수식 (4)는 속성 A의 값을 알고 있음으로서 분기(branching)되어 얻게 되는 기대 감소량을 나타낸다.

$$Gain(A) = I(s_1,s_2,...,s_m) - E(A) \text{ [8]} \cdots\cdots\cdots (4)$$

이렇게 각각의 속성에 의한 정보 이득 값으로 분할하여 가장 높은 정보 이득을 가지는 속성을 루트 검사 속성으로 정하고, 노드의 모든 샘플들이 같은 클래스에 속할 때까지 재귀적으로 분할한다.

III. 의사결정 트리 기법을 이용한 그리드 자원선택 시스템

우리는 그리드 사용자에게 적합한 자원 목록을 제공하기 위해 그림 2와 같이 사용자와 그리드 미들웨어 중간에 위치하여 사용자의 데이터 및 요구사항을 분석하고 이에 적합한 그리드 자원들을 선택하여 해당 자원 목록을 사용자들에게 제공하는 의사결정 트리 기반의 그리드 자원선택 시스템을 제안한다.

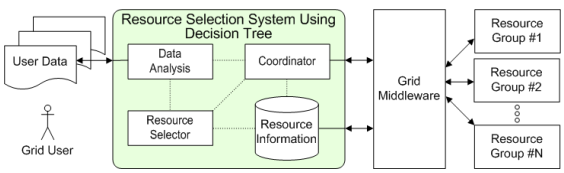


그림 2. 의사결정 트리 기반의 그리드 자원선택 시스템
Fig. 2. Decision Tree-based Grid Resource Selection System

3.1 자원 정보 모듈(Resource Information Module)

그리드 사용자의 데이터 특성에 맞는 자원을 제공하기 위해서는 자원들에 대한 정보 수집이 선행 되어야 한다. 자원 정보 모듈(Resource Information Module)은 기존의 그리드 미들웨어로부터 자원에 대한 정보 데이터를 수집한다. 수집된 데이터들은 데이터베이스에 저장하여 의사결정 트리를 생성하는데 사용된다. 본 논문에서 그리드 자원은 기본적으로 표 1과 같이 10개의 형태로 분류된다. 표 1은 CPU 속도, 메모리 크기, 저장용량과 같은 각 그리드 자원의 성능 수치에 따라 분류된 자원 형태($C_1 \sim C_{10}$)를 나타낸다. 또한, 각 그리드 자원은 작업 처리를 위한 대기 중인 Idle 상태와 작업을 처리 중인 Busy 상태로 구분된다.

표 1. 그리드 자원의 분류
Table 1. Classification of Grid Resource

Type	Resource Information		
	CPU Speed	RAM Size	Disk Space
C_1	700 Mhz	32 MByte	10 GByte
C_2	933 Mhz	128 MByte	20 GByte
C_3	1.0 Ghz	256 MByte	60 GByte
C_4	1.7 Ghz	384 MByte	120 GByte
C_5	2.0 Ghz	512 MByte	80 GByte
C_6	2.2 Ghz	512 MByte	80 GByte
C_7	2.4 Ghz	512 MByte	120 GByte
C_8	2.6 Ghz	1 GByte	120 GByte
C_9	2.8 Ghz	1 GByte	60 GByte
C_{10}	3.0 Ghz	2 GByte	250 GByte

자원선택 모듈(Resource Selection Module)로부터 가용 자원 요청 메시지를 수신 시 자원 정보 모듈은 현재 사용 가능한 후보 자원을 리스트 형태로 전송한다. 그 다음 자원선택 모듈이 사용할 자원을 선택하여 전송하면 자원 정보 모듈은 해당 자원의 현재 상태를 Idle 상태에서 Busy 상태로 변경하여 다른 사용자가 선택 할 수 없게 하는 역할을 하며, 반대로 사용중이었던 자원이 작업 처리를 완료한 후 유휴 상태가 되면 중계 모듈(Coordinator Module)로부터 해당 자원 정보를 수신받아 상태 정보 값을 Idle 상태로 변경하는 역할을 수행한다.

3.2 데이터 분석 모듈(Data Analysis Module)

데이터 분석 모듈(Data Analysis Module)은 사용자의 그리드 자원 사용 요청 메시지를 전달받아 데이터의 특성이나 사용자의 요구 사항을 추출한다. 이 모듈은 의사결정 트리의 구축을 위해 데이터 특성 및 사용자 요구사항과 그에 따라 선택 가능한 자원들을 학습 데이터 셋 형태로 저장한다. 본 논문에서는 사용자의 데이터 특성과 요구 사항 정보를 RealTime, DataSize, Distributed, Cost, Deadline 속성으로 나타내었다. 분석을 용이하게 하기 위해 모든 데이터는 정수로 표현되며 분석에 사용된 데이터 셋의 예는 표 2와 같다.

표 2에서 자원의 순서번호(Seq)를 제외한 모든 데이터는 1에서 9 사이의 정수로 기록되며 9에 가까울수록 해당 속성의 특징을 강하게 반영한다는 것을 표현한다. 예를 들어 RealTime은 사용자의 데이터가 실시간 데이터 인지를 나타내는 것으로 9에 가까운 값이면 빠른 데이터 처리를 요구한다. Distributed는 사용자의 데이터가 분산처리 환경에 용이한 것이지를 나타내며, DataSize는 처리 할 데이터의 크기를 나타낸다. Cost는 사용자가 그리드 자원을 사용함에 있어서 지불 할 수 있는 예산의 크기를 나타내며 큰 값일수록 예산이 충분한 것을 나타낸다. Deadline은 사용자가 요청한 데이터 처리종료 시간준수의 엄격함을 나타낸다.

표 2. 데이터 특성과 요구사항에 따른 자원선택의 예
Table 2. Examples for Resource Selection with data properties and user requirements

Seq	RealTime	DataSize	Distributed	Cost	Deadline	Resource
1	6	4	2	3	5	C_3
48	8	2	9	4	8	C_4
...	...	...	...	...	...	...
100	9	8	3	8	7	C_5

우리는 학습 데이터 셋으로 저장된 표 1의 목록을 분석하기 위해 C4.5 알고리즘을 이용하여 의사결정 트리를 생성한다. 의사결정 트리를 생성하기 위해서는 우선적으로 최상위 노드(Root Node)가 되는 속성을 찾아야 한다. 이를 위해 각 속성의 엔트로피를 계산하여 정보이득(Information Gain) 값이 가장 큰 속성을 선택해야 하므로, 우리는 수식 (1)[8]로부터 샘플 데이터 분류에 필요한 기대 정보량(Expected Information)을 계산한다. 100개의 샘플 데이터를 분류하는데 필요한 기대 정보량은 수식 (5)의 결과 값인 3.242와 같다.

$$I(s_1, s_2, ..., s_{10}) = I(16,7,9,9,8,9,5,9,12,16) \quad \cdots\cdots\cdots\cdots (5)$$
$$= 3.242$$

다음으로 각 속성의 엔트로피를 계산한다. 예를 들어 RealTime 속성의 엔트로피를 계산하기 위해서는 각각의 데이터의 RealTime 속성 값이 어떤 클래스($C_1 \sim C_{10}$)에 속하는지 분포를 알아야 한다. 각 분포에 대한 기대 정보량을 계산하여 수식 (6)[8]의 결과 값들을 얻을 수 있다.

$$For\ Real\ Time < 5 \quad \cdots\cdots (6)$$
$$s_{11} = 16, ..., s_{101} = 0 \;\; I(s_{11}, ..., s_{101}) = 2.253$$
$$For\ Real\ Time \geq 5$$
$$s_{12} = 0, ..., s_{102} = 16 \;\; I(s_{12}, ..., s_{102}) = 2.228$$

그 다음 수식 (2)[8]를 이용하여 샘플들이 RealTime 속성으로 분할될 경우 주어진 샘플을 분류하는데 요구되는 기대 정보량을 계산한다. 계산된 기대 정보량은 수식 (7)의 결과인

2.24와 같다.

$$E(RealTime) \quad \cdots\cdots (7)$$
$$= \frac{49}{100} I(s_{11}, ..., s_{101}) + \frac{51}{100} I(s_{12}, ..., s_{10,2})$$
$$= 1.10397 + 1.13628 \fallingdotseq 2.24$$

수식 (4)[8]를 이용하여 수식 (7)의 결과로부터 RealTime으로 분할될 경우의 정보 이득인 수식 (8)의 결과 1.002를 구할 수 있다.

$$Gain(RealTime) = I(s_1, s_2, ..., s_m) - E(A) \quad \cdots (8)$$
$$= 3.242 - 2.24 = 1.002$$

수식 (5)에서 수식 (8)까지 계산 했던 방법과 동일하게 다른 속성에 대한 정보 이득을 계산하면 Gain(DataSize) = 0.992, Gain(Distributed) = 0.884, Gain(Cost) = 0.956, Gain(Deadline) = 0.884를 얻을 수 있다. 이들 중 가장 높은 정보 이득을 가지는 속성이 RealTime이므로 RealTime 을 검사 속성으로 선택한다. 즉, RealTime 이라는 레이블의 새로운 루트 노드(Root Node)가 생성이 되고 가지들이 각각의 속성 값들에 대해서 생성된다. C4.5 알고리즘[9]을 재귀적으로 수행하면 각 속성은 그림 3과 같이 분할되어 의사결정 트리를 만들 수 있다.

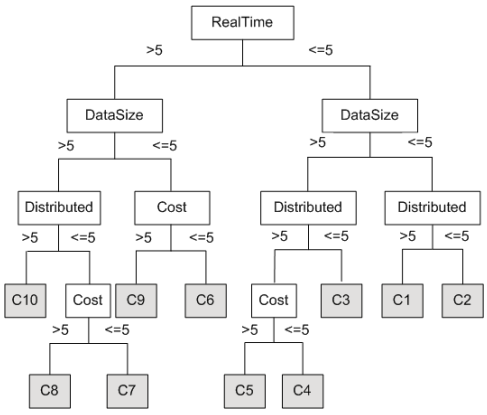


그림 3. 자원선택을 위한 의사결정 트리 구조
Fig. 3. Structure of Decision Tree for Resource Selection

그림 3에서 회색으로 표시한 노드는 분할이 완료된 단말 노드(Leaf Node)들을 나타내며 단말 노드 전 단계의 모든 노드들은 각 단계에서 분할 될 때 사용되는 검사 속성들을 나타낸다. 부등호 표시의 숫자들은 속성 값이 속하는 조건을 나태내고 있다.

3.3 자원선택 모듈(Resource Selection Module)

자원선택 모듈(Resource Selection Module)은 데이터 분석 모듈로부터 그리드 사용자의 자원 사용 요청 메시지와 함께 분석이 완료된 데이터를 수신한다. 이 모듈은 데이터 분석 모듈로부터 그리드 자원 사용요청 메시지를 전송받으면 자원 정보 모듈에게 가용 자원 요청 메시지를 송신하고, 자원 정보 모듈이 전송한 후보 자원 리스트를 수신한다.

자원선택 모듈은 분석된 그리드 사용자의 데이터와 자원 정보 모듈로부터 수신한 사용 가능 후보 자원들 중, 100 개의 학습 데이터 셋으로 분류되어 구축된 그림 3의 의사결정 트리를 이용하여 선택 조건에 적합한 다수개의 자원을 선택하여 자원 목록을 생성한다. 예를 들어, 어떤 사용자의 데이터가 RealTime = 9, DataSize = 8, Distributed = 3, Cost = 7, DeadLine = 8 이라는 속성 값을 가지고 있다고 가정하자. 그림 3의 의사결정 트리로부터 루트 노드의 검사 속성 RealTime이 5보다 큰 값인 9이므로 좌측 자식 노드(Left Child Node)인 DataSize로 이동하고 이 역시 5보다 큰 8의 값이므로 좌측 자식 노드인 Distributed로 이동한다. 다른 속성에 대해도 같은 방법으로 트리를 탐색하여 단말 노드에 도달하면 이 데이터에 적합한 자원의 형태는 C_8이고, 이 형태에 매칭 되는 자원들을 선택한다.

선택된 자원은 현재 해당 자원을 가용 자원 목록에서 제외하고 실제 사용자의 데이터를 처리하기 위해 자원 정보 모듈과 중계모듈(Coordinator Module)로 보낸다.

3.4 중계 모듈(Coordinator Module)

중계 모듈은 자원선택 모듈로부터 전달 받은 자원들의 스케줄링을 담당하는 모듈로써 작업의 분배 및 할당 기능을 제공한다. 중계 모듈은 선택된 자원들에 대한 성능 측정을 위한 다양한 스케줄링 기법들을 제공한다. 본 논문에서 사용되는 스케줄링 기법은 선택한 자원에 순차적으로 작업을 분배하는 Round-robin(RR) 방식, 임의의 순서대로 분배하는 Random(RA) 방식, 자원의 신뢰성을 측정하여 높은 신뢰성을 가진 자원에게 할당하는 Reliability-Measurement System(RM)[11]이다.

또한, 중계 모듈은 작업의 처리가 완료된 자원들로부터 결과를 취합하여 파일로 저장하고 그리드 사용자에게 반환하는 역할을 한다. 이 때 Busy 상태였던 작업이 끝난 자원을 다시 Idle 상태로 변경하기 위해서 해당 자원 목록을 자원 정보모듈로 전송한다.

Ⅳ. 실험 및 결과 분석

우리는 의사결정 트리 기법을 이용한 그리드 자원선택 시스템 (RSSUDT: Resource Selection System Using Decision Tree)의 자원선택 적합성 및 효율성을 평가하기 위해 DEVS-JAVA[12] 기반의 시뮬레이션 모델을 구현하고 기존 그리드 자원선택 시스템인 Condor-G 및 Nimrod-G와 비교하였다. 각 그리드 자원선택 시스템은 그리드 사용자의 요구사항과 데이터 특성에 따라 고유의 자원선택 방법을 이용하여 10개의 자원을 선택하였다. 그 다음 우리는 앞서 언급한 RR, RA, RM과 같은 다양한 스케줄링 기법을 이용하여 선택된 10개의 자원에게 작업을 할당함으로써 각각의 작업 처리율, 자원 활용률, 작업 손실 및 평균 작업 처리 시간을 측정하였다.

4.1 의사결정 트리 기법을 이용한 그리드 자원선택 시스템의 시뮬레이션

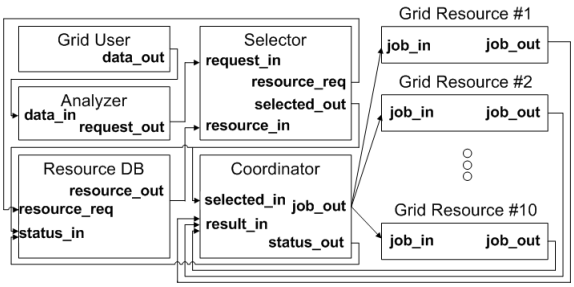


그림 4. DEVSJAVA-based 기반의 RSSUDT 시뮬레이션 모델
Fig. 4. DEVSJAVA-based RSSUDT Simulation Model

성능 평가를 위해 구현된 의사결정 트리 기반 그리드 자원선택 시스템의 시뮬레이션 모델은 그림 4와 같이 5가지 형태의 컴포넌트들로 구성된다.

그림 4에서 Grid User는 그리드 사용자로서 처리할 데이터를 그리드 자원선택 시스템에 요청한다. Analyzer는 그리드 사용자의 자원 요청 메시지를 받아 응답하고, 100개의 학습 데이터 셋으로 완성된 의사결정 트리 정보를 포함하고 있으며, 사용자의 데이터가 어떤 속성을 나타내는지 추출해서 Selector로 보낸다. Selector는 Analyzer로부터 자원 사용 요청 메시지와 분석된 데이터를 전달 받으면 Resource DB에게 가용 자원 목록 요청 메시지를 보내고 후보자원을 전달받아, 후보자원 중 완성된 의사결정 트리를 이용하여 10개의 자원을 선택한다. 이때 선택한 자원 목록을 Resource DB와 Coordinator에 동시에 보낸다. Resource DB는 그리드 환경에서 자원에 대한 정보를 저장하고 있는 데이터베이스로서 Selector의 가용자원 요청 메시지에 응답하여 현재 사용 가능한 후보자원 목록을 전송한다. 또한, Selector가 보낸 현재 선택된 자원은 Idle 상태를 Busy 상태로 전환되어 가용자원 목록에서 제외되고, Coordinator가 보낸 작업 완료 자원 목록에 대해서는 다시 가용 상태로 변경한다. Coordinator는 RR, RA, RM 스케줄링 기법을 이용하여 Selector로부터 전달받은 선택된 자원들에게 작업을 분배 및 할당하고, 작업 처리가 완료된 자원을 다시 가용한 자원 상태로 만들기 위해 Resource DB에 전달한다.

본 실험에서 하나의 그리드 사이트(Grid Site)는 CPU 속도, 메모리 크기, 디스크 용량 등을 무작위로 생성된 1000개의 가용자원을 포함한다. 비슷한 성능을 가지는 자원을 크게 10개로 나누어 클래스로 구분한다. 데이터의 속성과 사용자의 요구사항에 의해 선택할 수 있는 요소는 표 1과 같으며 100개의 학습 데이터 셋을 C4.5 알고리즘을 이용하여 Analyzer가 의사결정 트리를 구축한다. RSSUDT에서는 Selector가 이 의사결정 트리로 부터 사용자의 요구사항과 데이터 특성에 대응하는 10개의 자원을 선택한다. Condor-G는 자원을 선택함에 있어서 원하는 자원 조건을 상세히 명세 할 수 있지만, 자원이 광고한 내용을 완전히 신뢰할 수 없고, 하나의 자원만 관리되므로 일정한 성능 이상인 것을 선택하였다. Nimrod-G는 빠른 데이터 처리를 요하므로 표 2의 Deadline 속성이 높은(H) 값으로 선택하였다. 이렇게 선택된 자원을 Coordinator가 RR, RA, RM 방식으로 스케줄링한 후 일정 시간 간격마다 성능 지표를 측정한다.

4.2 실험 결과

첫 번째 실험에서는 Condor-G, Nimrod-G, RSSUDT에 의해 선택된 자원들에 RR, RA, RM 스케줄링 기법들을 이용하여 작업 할당한 후 작업 처리율을 측정하였다. 그림 5~7은 각 스케줄링 기법에 따른 Condor-G, Nimrod-G, RSSUDT의 작업 처리율의 변화를 나타내고 있다. Condor-G의 경우 RR, RA, RM 스케줄링 방식에 의해 측정된 각각의 평균 작업 처리율은 0.3326, 0.3331, 0.3827이었고, Nimrod-G의 경우는 0.2994, 0.3012, 0.3151이었으며 RSSUDT는 0.3661, 0.3633, 0.4203이었다. 세 가지 스케줄링 기법 모두에서 RSSUDT에 의해 선택된 자원들을 이용하여 작업을 처리한 결과가 다른 모델들과 비교하여 더 나은 작업 처리율을 보임을 알 수 있다. 작업 처리율은 단위 시간당 처리 한 작업의 개수로서 이기종의 자원으로 구성된 그리드 컴퓨팅 환경에서 빠른 작업을 요하는 작업에 대하여

본 논문에서 제안된 시스템이 보다 더 나은 자원을 적당하게 선택함을 알 수 있다.

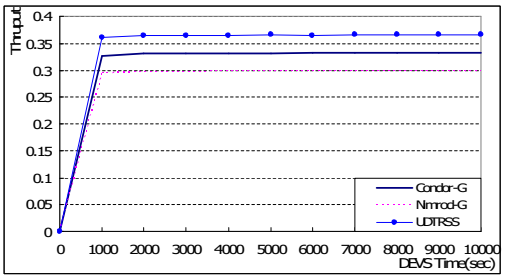


그림 5. RR(Round-robin) 스케줄링의 작업 처리율
Fig. 5. Throughput of RR(Round-robin) Scheduling

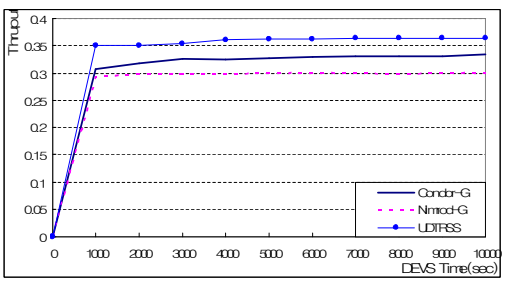


그림 6. RA(Random) 스케줄링의 작업 처리율
Fig. 6. Throughput of RA(Random) Scheduling

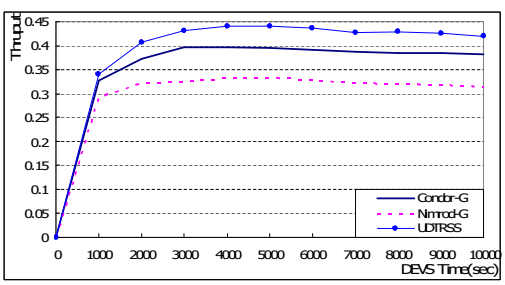


그림 7. RM(Reliability-Measurement) 스케줄링의 작업 처리율
Fig. 7. Throughput of RM(Relaibility-Measurement) Scheduling

두 번째 실험은 Condor-G, Nimrod-G, RSSUDT의 자원 활용률을 측정하기 위한 실험이다. 자원 활용률[11]은 수식 (9)를 이용하여 계산 가능하며 여기서 n은 총 자원의 개수, R_n은 n번째의 자원, SJ_n은 n번째 자원의 처리된 작업의 개수, PT_n 은 n번째 자원의 처리 시간을 나타내며 ST은 시뮬레이션 시간(Simulation Time)을 의미한다.

$$Utilization(\%) = \frac{(\sum_{i=1}^{n} R_n \times SJ_n) \times PT_n}{ST} \times 100 [11] \cdot \cdot (9)$$

그림 8~10은 각 스케줄링 기법에 따른 Condor-G, Nimrod-G, RSSUDT의 자원 활용률의 변화를 보여주고 있다. 각 그림을 보면 알 수 있듯이 RR, RA, RM 스케줄링 기법에 따른 Condor-G에 의해 선택된 자원들의 평균 활용률은 66.546, 66.633, 76.585이었고, Nimrod-G는 59.880, 60.264, 63.070이었으며 RSSUDT는 73.234, 72.660, 84.093의 자원 활용률을 기록했다. 자원 활용률 역시 작업 처리율과 마찬가지로 RSSUDT에 의해 선택된 자원에 작업을 할당하였을 경우 모든 스케줄링 기법에서 다른 자원선택 시스템보다 더 나은 성능을 보였으며 이는 그리드 자원을 효율적으로 사용한다는 것을 알 수 있다. RM 스케줄링 기법을 적용하여 자원 활용률을 측정한 경우 그림 10과 같이 5000초 이후부터는 모든 자원선택 시스템의 자원 활용률이 감소함을 알 수 있는데, 이는 RM 스케줄링 기법이 가장 높은 신뢰성을 갖는 자원에게 작업을 할당하는 방법이므로 5000초 이상부터는 가장 높은 신뢰성을 갖는 자원이 처리하고 저장 할 수 있는 한계를 넘어섰기 때문이다. 하지만 전체 자원 활용률은 RR과 RA에 비하여 높은 수치를 기록했다.

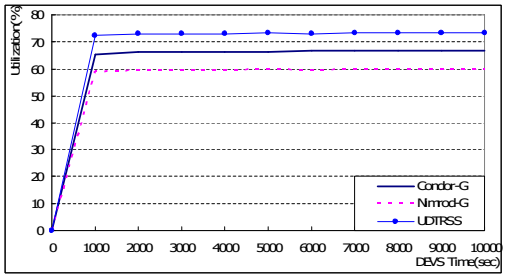


그림 8. RR(Round-robin) 스케줄링의 자원 활용률
Fig. 8. Utilization of RR(Round-robin) Scheduling

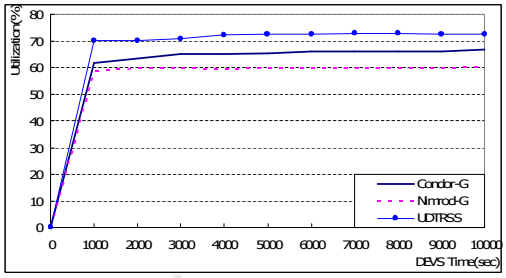


그림 9. RA(Random) 스케줄링의 자원 활용률
Fig. 9. Utilization of RA(Random) Scheduling

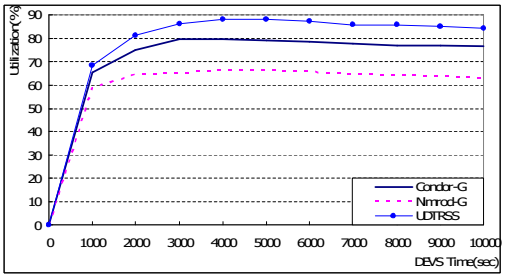


그림 10. RM(Reliability-Measurement) 스케줄링의 자원 활용률
Fig. 10. Utilization of RM(Relaibility-Measurement) Scheduling

세 번째 실험은 Condor-G, Nimrod-G, RSSUDT에 의해 선택된 자원들에 각 스케줄링 기법들을 이용하여 작업 할당할 경우 발생되는 작업 손실의 측정을 위한 실험이다. 그림 11~13은 각 스케줄링 기법에 따른 Condor-G, Nimrod-G, RSSUDT의 손실된 작업 개수를 나타낸다. Condor-G는 RR, RA, RM 스케줄링 기법들을 순서대로 적용하였을 경우 각각 1674, 1669, 1173개의 작업 손실을 기록하였으며 Nimrod-G는 각각 2006, 1988, 1849개의 작업 손실을 기록하였다. RSSUDT의 경우 다른 두 시스템보다 현저히 낮은 1339, 1367, 797개의 작업 손실을 기록하였다. 이는 제안된 시스템이 다양한 자원이 존재하는 그리드 컴퓨팅 환경에서 보다 더 안정적으로 작업을 처리 할 수 있도록 더 나은 그리드 자원을 선택하여 그리드 자원의 신뢰성을 높여주고 그리드 전체 환경에서의 작업 손실로 인한 재분배 및 할당, 재처리에 드는 시간과 비용을 절감하는 효과를 제공한다는 것을 증명한다.

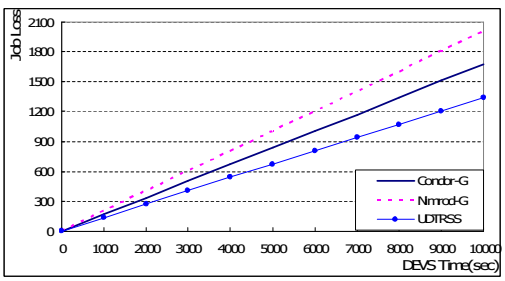


그림 11. RR(Round-robin) 스케줄링의 작업 손실
Fig. 11. Job Loss of RR(Round-robin) Scheduling

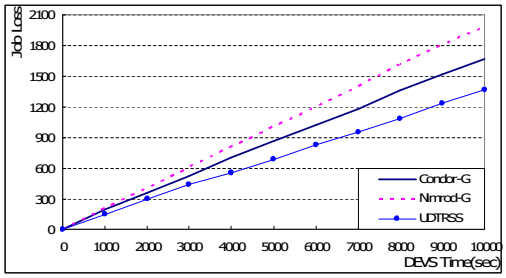


그림 12. RA(Random) 스케줄링의 작업 손실
Fig. 12. Job Loss of RA(Random) Scheduling

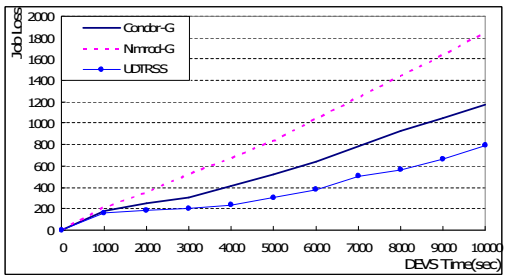


그림 13. RM(Reliability-Measurement) 스케줄링의 작업 손실
Fig. 13. Job Loss of RM(Relaibility-Measurement) Scheduling

마지막 실험은 Condor-G, Nimrod-G, RSSUDT에 의해 선택된 자원들에 각 스케줄링 기법들을 이용하여 작업 할당할 경우 소요되는 평균 작업 처리 시간을 측정하기 위한 실험이다. 평균 작업 처리 시간은 요청한 작업이 처리되어 반환되기까지 소요되는 평균 시간을 의미한다. 그림 14~16은 각 스케줄링 기법에 따른 Condor-G, Nimrod-G, RSSUDT의 자원 활용률의 변화를 보여주고 있다. RR, RA, RM 스케줄링 기법들을 순서대로 적용하였을 경우 Condor-G는 670.609, 676.879, 766.717의 평균 작업 처리 시간을 기록하였고, Nimrod-G는 620.072, 618.945, 812.168의 평균 작업 처리 시간을 기록하였으며, RSSUDT는 434.275, 444.123, 630.378의 평균 작업 처리 시간을 기록하였다. 여기서 RR이나 RA에 비하여 RM의 평균 작업 처리 시간이 높은 이유는 RR이나 RA에서는 자원의 저장 용량이나 메모리 크기에 관계없는 순차적 혹은 임의적 작업 할당에 의한 작업 손실이 발생하였으나, RM은 신뢰성을 바탕으로 작업을 할당하므로 빠른 처리가 가능하고 저장 매체 용량이 큰 자원에게 작업을 할당하여 상대적으로 적은 량의 작업이 손실되었기 때문이다. 즉, 소요 시간에 작업이 큐에 머무르는 시간을 더해지므로 전체적으로 평균 작업 처리시간 역시 증가한 것이다. 앞의 세 실험에서 Condor-G가 Nimrod-G

보다 더 나은 성능을 보였으나 평균 작업 처리 시간이 증가한 이유도 이와 같은 맥락으로 설명 할 수 있다.

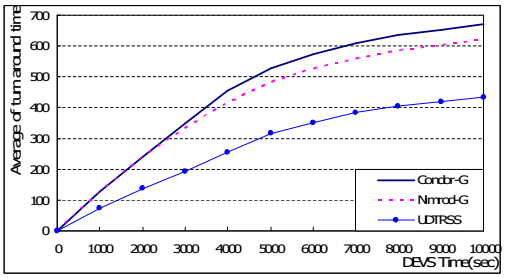


그림 14. RR(Round-robin) 스케줄링의 평균 작업 처리시간
Fig. 14. Average of Turn Around Time of RR(Round-robin) Scheduling

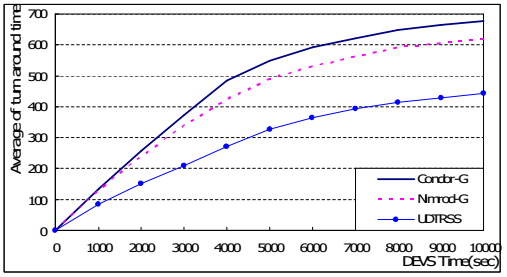


그림 15. RA(Random) 스케줄링의 평균 작업 처리시간
Fig. 15. Average of Turn Around Time of RA(Random) Scheduling

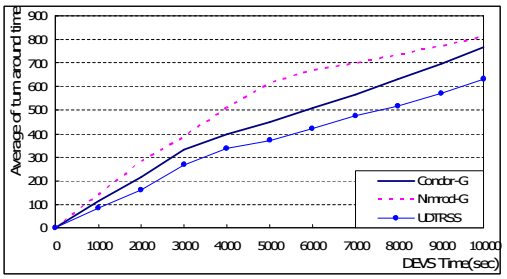


그림 16. RM(Reliability-Measurement) 스케줄링의 평균 작업 처리시간
Fig. 16. Average of Turn Around Time of RM(Relaibility-Measurement) Scheduling

만약 Condor-G의 경우, 최고의 성능을 가지는 자원을 전부 하나씩 선택하였다면 Condor-G의 모든 성능 지표는 최고치를 넘어서 완전무결한 결과 값을 보였을 것이다. 즉, 자원 활용률 = 100%, 작업 손실 = 0 등등의 값을 기록 했을 것이다. 하지만 이것은 자원이 광고하는 내용이 실제와 다르면 이 같은 수치는 기록 할 수 없고, 실제 그리드 환경과는 거리가 있으므로 RSSUDT 보다 낮은 성능 수치를 기록한 것으로 보이며, Nimrod-G의 결과도 같은 맥락으로 이해할 수 있다. 실험 결과에서 보듯이 비교적 정확하게 분류된 자원 정보의 데이터베이스를 갖는 RSSUDT에 의해 선정된 자원들로 작업을 분산 처리하였을 때의 결과가 기존의 그리드 자원선택 시스템인 Condor-G 또는 Nimrod-G로 자원을 선택하였을 경우보다 더 나은 시스템 성능을 제공할 수 있다. 이는 본문에서 제안한 의사결정 트리 기반의 그리드 자원선택 시스템이 그리드 사용자로부터의 고성능의 그리드 자원을 요구 시 적합한 자원을 단 시간 내에 효율적으로 선택한다는 것을 증명하고 있다.

V. 결론 및 향후 과제

본 논문은 그리드 컴퓨팅 환경에서 대용량 분산 작업 처리를 위해 그리드 사용자의 요구사항을 충족시키고 처리 요청된 데이터의 특성에 적합한 그리드 자원을 선택하여 제공하기 위해 의사결정 트리 기법을 이용한 그리드 자원선택 시스템을 제안하였다.

이 시스템은 그리드 자원의 특징을 분석하여 데이터 셋으로 저장하고, 사용자가 처리 할 데이터의 속성과 사용자의 요구 사항으로 선택하는 자원을 학습데이터 셋으로 저장하고 분류하여 자원선택에 필요한 의사결정 트리를 구축한다. 자원 요청 시 마다 자원 분류, 탐색, 선택에 많은 시간을 소모하는 기존의 그리드 자원선택 시스템들과는 달리 본 논문에서 제안된 시스템은 그리드 사용자의 자원 사용 요청 시 의사결정 트리의 간단한 탐색 과정을 통해 사용자가 원하는 그리드 자원을 빠른 시간 내에 자동적으로 탐색 및 선택할 수 있다.

실험 결과를 통해 본 논문에서 제안된 의사결정 트리 기법을 이용한 그리드 자원선택 시스템인 Nimrod-G와 Condor-G와 비교하여 더 높은 작업 처리율, 자원 활용률과 낮은 작업 손실, 처리 시간을 제공함으로써 사용자 요구사항 및 데이터 특성에 적합한 그리드 자원을 효율적으로 선택하여 작업을 분산 처리하는 데 효과적이라는 사실을 알 수 있었다. 실험에서는 고성능, 고용량의 자원을 요구할 것이라고 한정지었지만 그렇지 않은 경우라도 적합한 학습 데이터 셋으로 정확하게 분류한다면 그리드 사용자에게 적당한 자원을 선택할 것으로 예상한다. 향후 연구에서는 데이터의 속성과 사용자의 요구 사항을 반영하여 분류하기 위해 더욱 더 다양한 측정 변수를 입력하고, 일정 수 이상의 데이터 셋이 수집 시 의사결정 트리를 재구성 하여 주기적으로 사용자의 자원선택을 반영함으로써 자원선택의 적합성 및 정확성을 능동적으로 개선시킬 수 있는 그리드 자원선택 시스템을 개발할 예정이다.

참고문헌

〔1〕 Rajkumar Buyya, David Abramson, and Spikumar Venugopal. "The Grid Economy", Proceedings of the IEEE, VOL. 93, NO. 3, 2005.

〔2〕 김정엽. "의사결정 트리를 이용한 인천시 도시성정 예측 모델링", 석사학위논문, 인하대학교, 2004.

〔3〕 마용범, 이종식. "사용자 요구기반의 그리드 거래 관리 모델", 한국시뮬레이션학회 논문지, Vol.15, No.3, pp. 11-21, 2006.

〔4〕 Sung Ho Jang and Jong Sik Lee. "Service Agent-Based Resource Management Using Virtualization for Computational Grid", Lecture Notes in Computer Science, Vol. 4488, pp.966-969, 2007.

〔5〕 노남수, 홍필두, 장택수, 이용우. "Grid환경기반 자원 선택에 관한 연구", 한국컴퓨터종합학술대회, Vol. 32, No. 1(A), 2005.

〔6〕 이준돈, 정윤미, 길아라, 윤현주. "그리드 컴퓨팅 환경에서 효율적인 자원 활용을 위한 성능 계량 모델 및 자원 선택 알고리즘 제안", 한국정보과학회 가을 학술발표논문집, Vol.30, No.2, 2003.

〔7〕 R. Buyya. "Economic-based Distributed Resource Management and Scheduling for Grid Computing", http://www.buyya.com/thesis/, 2002.

〔8〕 Jiawei Han and Micheline Kamber. "Data mining : concepts and techniques", Morgan Kaufmann Publishers, 2001.

〔9〕 이배희. "위치정보와 의사결정 트리 기법을 이용한 개인화된 추천 시스템", 석사학위논문, 인하대학교, 2006.

〔10〕 이극노. "이동통신고객 분류를 위한 의사결정나무(C4.5)와 신경망 결합 알고리즘에 관한 연구", 한국지능정보시스템학회논문지, 제9권 제1호, pp.139-155, 2003.

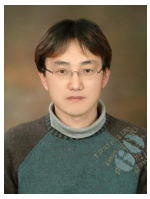
〔11〕 박다혜. "자원 신뢰성 측정을 통한 효율적인 그리드 자원 스케줄링 모델", 한국시뮬레이션학회 논문지, Vol. 15, No. 2. pp.129-9, 2006.

〔12〕 B.P. Zeigler, et al. "DEVS Framework for Modeling, Simulation, Analysis and Design of Hybrid systems in Hybrid II", Lecture Notes in CS, Vol. 3045 Springer-Verlag, Berlin, 1997.

저 자 소 개

노 창 현

2006 수원대학교 인터넷정보공학과 학사
2006 .8~현재 인하대학교 정보공학과 컴퓨터정보공학 석사과정
관심분야 : 그리드 컴퓨팅, 모델링 및 시뮬레이션, 임베디드 시스템



조 규 철

2005 인하대학교 컴퓨터공학부 학사
2007 인하대학교 정보공학과 컴퓨터정보공학 석사
2007. 3~현재 인하대학교 정보공학과 컴퓨터정보공학 박사과정
관심분야 : 그리드 컴퓨팅, 소프트웨어 공학, 패턴인식

마 용 범

2005 인하대학교 컴퓨터공학부 학사
2007 인하대학교 정보공학과 컴퓨터정보공학 석사
2007. 3~현재 인하대학교 정보공학과 컴퓨터정보공학 박사과정
관심분야 : 그리드 컴퓨팅, 모델링 및 시뮬레이션

이 종 식

1993 인하대학교 전자공학과 학사
1995 인하대학교 전자공학과 석사
2001 미국 애리조나대 전기·컴퓨터공학과 박사
2001~2002 캘리포니아 주립대학교 전기·컴퓨터공학과 전임강사
2002~2003 클리블랜드 주립대학교 전기·컴퓨터공학과 조교수
2003~2006 인하대학교 컴퓨터 공학부 조교수
2006~현재 인하대학교 컴퓨터 공학부 부교수
관심분야: 시스템 모델링 및 시뮬레이션, 그리드 컴퓨팅